**Project Proposal for ECE 7502 Class**

**Spring 2015**

**Topic: Canary SRAM Built in Self-Test for SRAM VMIN Tracking**

**Student: Arijit Banerjee**

**Actual Date: 2/17/2015**

**Revised Date: 2/19/2015**

**Introduction:** With the technology scaling in deep submicron technologies, SRAM write ability, readability and read stability degrades. However, SRAM write operation is affected the most [1] which creates a bottleneck in further SRAM minimum operating voltage (VMIN) scaling and have more energy savings using state of the art dynamic voltage and frequency scaling (DVFS) [2] technique. We can use the dual rail [3] techniques to lower the peripheral supply voltage, but retain the SRAM core voltage at a higher voltage to reduce read stability related issues. However, this dual rail technique is costly from design and implementation standpoint. On the other hand, we can use a peripheral assist circuit to overcome this bottleneck and lower the SRAM VMIN further. However, this assist approach comes with a constant energy and area overhead. Moreover, the SRAM write VMIN changes with respect to process, voltage and temperature (PVT) conditions. In addition, the SRAM ages with time while in usage and the SRAM VMIN gets higher. Hence, we need to track the SRAM VMIN across PVT conditions using a closed loop system. Canary circuits [4] are not new and authors have demonstrated canary circuits in SRAM, which we describe in the next section.

**Prior Works in Canary SRAMs:** Authors in[5] show that we can track SRAM data retention voltage (DRV) using a canary SRAM. The paper [6] show that we can use a smaller SRAM as canary SRAM, to track the core SRAM dynamic write VMIN using reverse assist [6] techniques. We can use the canary SRAMs to capture a bit failure rate (BFR) for a certain reverse assist to compare with a failure threshold rate so that we can take actions in case the canary BFR changes. The set of actions can be in terms of turning on or off assist or inform the DVFS controller to increase the supply voltage or reduce the clock frequency as required. However, no research has been done in the domain of testing of the canary SRAM for a valid targeted BFR. Authors in [6] show that the canary SRAM using reverse assist demonstrate a unique failure probability vs. reverse assist curves. Therefore, we can expect this unique relationship in silicon results too. Hence, in this class project we propose a canary built in self-test (CBIST) hardware to test the canary SRAM for specific BFRs corresponding to different reverse assist condition. Next section describes the common fault models available for SRAM testing.

**SRAM Faults:** We can classify the SRAM faults broadly in two categories: static faults and dynamic faults. In static faults, we can have multiple fault types like stuck at faults (SAF), address decode faults (AF), transition faults (TF) etc. On the other hand, in the dynamic fault category we can have faults like recovery faults and disturb faults. Examples of recovery faults can be sense-amplifier recovery fault such as saturation after long time reading and writing 0 or 1s, and write recovery fault as write followed by read or write at different location resulting in reading or writing at the same location etc. In addition, dynamic faults can occur due to high-speed operation of the SRAMs that can lead to functional faults. Next section describes the state of the art built in self-test (BIST) algorithms for SRAM testing.

**State of the Art SRAM BIST Algorithms for Testing:** SRAM testing without BIST can take huge amount of time for a bigger SRAM size. Usually, we do SRAM testing using MARCH tests; those are simple sequences for write and read operations.For SRAM testing using BIST, traditionally, we use various MARCH algorithms as per SRAM test-coverage requirements.Most of the initial MARCH algorithms were invented from 1960-80s and much prior work exists in this field. Modern trends for SRAM BIST implementations are more about programmable BISTs [7-10]; however, there is no relevant publications published for canary BISTs (CBIST). Hence, we propose this novel CBIST project to investigate how a canary BIST can improve testing in canary SRAMs.

**Canary BIST Challenges:** A canary SRAM for tracking SRAM write VMIN should have a certain bit failure rate (BFR) for a certain applied degree of reverse assist. It is a challenging task to identify if the canary SRAM BFR is within a valid range. In order to verify if the BFR for the canary SRAM, we need to retrieve this failure rate information by the canary BIST at the time of manufacturing and compare with a maximum and a minimum failure rate to see if the canary SRAM is behaving as expected. We can decide to pass the chip, if the canary SRAM BFR lies within the valid range measured using the CBIST, else we reject the chip. On the other hand, there are many possible types of reverse assists like WL under drive [11], positive bitline [6], negative VSS [11], VDD boost [11] for write operation. Hence, we need to make sure to provide enough hardware support on the CBIST side to do the canary SRAM tests quickly. However, we also need to make sure that the area cost of the CBIST is not increasing too much. In addition, we need to run the CBIST at speed to capture the exact functionality of the canary SRAM using different available reverse assists for a particular supply voltage.

On the other hand, the use of canary SRAM is a bit different from user perspective from system on chip (SoC) standpoint. The user needs to compute the bit error rate for a fixed reverse assist continuously to capture increase or decrease (exception) in bit failure rate compared to the specified rate to take an action. The action can be in the form of turning on or off assist for the SRAMs sharing the same power rail as the canary SRAM. In addition, if an exception occurs, a user can also talk to the DVFS unit and raise the supply voltage or lower the clock frequency as necessary. Hence, we provide the main requirements and challenges of canary BIST below in bullets:

* Canary BIST requirements
  + Manufacturing test for canary SRAM and BIST itself
  + At speed test for supporting user requirements in an system on chip (SoC)
* Challenges
  + At speed continuous testing for canary failures during manufacturing and user run conditions for different reverse assist and degree of reverse assists
  + Compute failure rate within a few cycles
  + Alert the user for assist related changes
  + CBIST testing itself
  + Power challenges

**Approach: Proposed Canary BIST (CBIST):** We propose the following algorithm to implement in the CBIST as follows. The CBIST will support a BIST algorithm to compute the number of bit failure in the canary SRAM for a specific reverse assist and a specific supply voltage and compare this value with a known upper and lower bound values specified by a user. We will implement the CBIST using the following basic components as follows:

* A state machine to write and read and compute the BFR of the canary SRAM
* A table for user specified valued of maximum BFR (MXBFR) and minimum BFR (MNBFR)
* A table for CBIST to store the computed BFR values for different reverse assists
* Two comparators to compare the MXBFR and MNBFR value to the computed canary BFR values for a reverse assist setting
* A control logic block to control all the blocks in the CBIST

The CBIST state machine will try to initialize the canary SRAM using a known complement of a word and then will write the word at the same address location. Thereafter it will read the word back, and a bitwise XOR block will check how many write failure has occurred. The next stage will be a bit-error-accumulator, which will add up the total number of write bit failure in the entire canary SRAM for a specific reverse assist. Here, we assume that we will integrate the reverse assists in the canary SRAM and it will have a maximum of eight different values to apply. Hence, the maximum number of bits required to control a single reverse assist are three bits. These three control bits of the reverse assist will not only control the degree of reverse assist in the canary SRAM, but also it will select a decoded MXBFR and MNBFR for a specific setting of reverse assist to compare with. We plan to test one reverse assist at a time for the canary SRAM and we will provide only two sets of reverse assist control bits for testing a maximum of two types of reverse assists. We will use these MXBFR and MNBFR values to compare the on the fly calculated BFR values of the canary SRAM using the CBIST and if the calculated BFR values are not in the range we will stop the CBIST and report as a failed test. On the other hand, if the test passes for one reverse assist setting the CBIST will try to test exhaustively all the reverse assist settings.

We will ensure that the CBIST will compute the BFR for the canary SRAM with a few cycles and will alert the user if canary BFR overshoot or undershoots the target BFR range. In addition, we will insert scan chains in the CBIST itself to test the CBIST itself and will report coverage using Synopsys TetraMAX tool. In addition, if time permits we will try to add clock gating and power gating features in the CBIST to have power management features.

**Important Design Metrics:** We define fault coverage of the CBIST, test-access-time (TAT) for the canary SRAM using CBIST, test-data-volume (TDV) of the canary SRAM as the three design metrics to quantify our approach. As we can bypass the traditional BIST MARCH tests for the canary SRAMs to an available SRAM BIST, we can save design cost, area and power for an additional implementation of features in canary BIST for fault coverage. However, we can implement certain type of MARCH algorithms using the CBIST hardware itself such as the sequence {write0, write1, read1; write1, write0, read0}, which will guarantee certain static and dynamic faults that we need to analyze.

**Expected Outcomes:** We have the following expected outcomes in shown in bullets:

* Canary BIST RTL simulations for the implementation
* SPICE results for CBIST pass/fail test after synthesis
* SPICE results of test access time after synthesis
* Fault coverage result for the canary BIST itself using Synopsys TetraMAX

**Deliverables and Timeline:** The following table shows the deliverables planned with expected dates. We will try to follow the expected data, and if we miss it, we will update the actual data and status accordingly.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Deliverables** | **Expected Date** | **Actual Date** | **Status** | **Issues** |
| RTL for Canary BIST in Verilog and simulation results | 2/26/2015 |  | Planned |  |
| Synthesis using DC with IBM 130nm | 2/28/2015 |  | Planned to report on 3/3/2015 |  |
| SPICE Simulation results of CBIST pass/fail testing and test access time | 3/14/2015 |  | Planned |  |
| Canary BIST Fault coverage results | 3/21/2015 |  | Planned to report on 3/24/2015 |  |

**References**:

[1] A. Bhavnagarwala et al., Fluctuation limits & scaling opportunities for CMOS SRAM cells, in IEDM Tech. Dig., 2005, pp. 659-662, 2005.

[2] S. Herbert and D. Marculescu, 'Analysis of dynamic voltage/frequency scaling in chip-multiprocessors, in Proc. Int. Symp. Low Power Electron. Design (ISLPED), pp. 38-43, 2007

[3] J. Pille et al., Implementation of the CELL Broadband Engine in a 65nm SOI Technology Featuring Dual-Supply SRAM Arrays Supporting 6GHz at 1.3V, in IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers, pp. 322-323, 606, 2007.

[4] B. H. Calhoun and A. P. Chandrakasan, Standby power reduction using dynamic voltage scaling and canary flip-flop structures, in IEEE J. Solid-State Circuits, vol. 39, no. 9, pp. 1504-1511, 2004.

[5] J. Wang and B. Calhoun, “Canary replica feedback for near-DRV  
standby vdd scaling in a 90 nm SRAM,” in Proc. Custom Integrated  
Circuit Conf. (CICC ’07), Sep. 2007, pp. 29–32.

[6] Banerjee, A.; Sinangil, M.E.; Poulton, J.; Gray, C.T.; Calhoun, B.H., "A reverse write assist circuit for SRAM dynamic write VMIN tracking using canary SRAMs," Quality Electronic Design (ISQED), 2014 15th International Symposium on , vol., no., pp.1,8, 3-5 March 2014

[7] Fradi, A.; Nicolaidis, M.; Anghel, L., "Memory BIST with address programmability," *On-Line Testing Symposium (IOLTS), 2011 IEEE 17th International* , vol., no., pp.79,85, 13-15 July 2011

[8] Zarrineh, K.; Upadhyaya, S.J., "Programmable memory BIST and a new synthesis framework," *Fault-Tolerant Computing, 1999. Digest of Papers. Twenty-Ninth Annual International Symposium on* , vol., no., pp.352,355, 15-18 June 1999

[9] Kokrady, A.; Ravikumar, C.P.; Chandrachoodan, N., "Layout-Aware and Programmable Memory BIST Synthesis for Nanoscale System-on-Chip Designs," *Asian Test Symposium, 2008. ATS '08. 17th* , vol., no., pp.351,356, 24-27 Nov. 2008

[10] Ching-Hong Tsai; Cheng-Wen Wu, "Processor-programmable memory BIST for bus-connected embedded memories," *Design Automation Conference, 2001. Proceedings of the ASP-DAC 2001. Asia and South Pacific* , vol., no., pp.325,330, 2001

[11] B. Zimmer, S. O. Toh, H. Vo, Y. Lee, O. Thomas, K. Asanovic, and  
B. Nikolic, “SRAM assist techniques for operation in a wide voltage range  
in 28 nm CMOS,” IEEE Trans. Circuits Syst. II, vol. 59, no. 12, pp. 853–  
857, Dec. 2012.